COMPUTER SCIENCE
TECHNICAL REPORT SERIES

UNIVERSITY OF MARYLAND
1807-1856-1920

UNIVERSITY OF MARYLAND
COLLEGE PARK, MARYLAND
20742

TR-977
DAAG-53-76C-0138

November 1980

MULTI-RESOLUTION PIXEL LINKING FOR IMAGE SMOOTHING AND SEGMENTATION

Teresa Silberberg
Shmuel Peleg
Azriel Rosenfeld

Computer Vision Laboratory
Computer Science Center
University of Maryland
College Park, MD 20742

Technical rept,

ABSTRACT

When an image is smoothed using small blocks or neighborhoods, the results may be somewhat unreliable due to the effects of noise on small samples. When larger blocks are used, the samples become more reliable, but they are more likely to be mixed, since a large block will often not be contained in a single region of the image. A compromise approach is to use several block sizes, representing versions of the image at several resolutions, and to carry out the smoothing by means of a cooperative process based on links between blocks of adjacent sizes. These links define "block trees" which segment the image into regions, not necessarily connected, over which smoothing takes place. In this paper, a number of variations on the basic block linking approach are investigated, and some tentative conclusions are drawn regarding preferred methods of initializing the process and of defining the links, yielding improvements over the originally proposed method.

The support of the Defense Advanced Research Projects Agency and the U.S. Army Night Vision Laboratory under Contract DAAG-53-76-C-0138 (DARPA order 3206) is gratefully acknowledged, as is the help of Janet Salzman in preparing this paper. The authors also acknowledge the help of Tsai-Hong Hong in investigating the effects of the linking process on the histogram of the image.

411074

1. Introduction

Suppose that an image is composed of a few types of regions each having approximately constant gray level. In principle, the image can be segmented into these regions by gray level thresholding, i.e., by slicing the grayscale into intervals, and classifying each pixel according to the interval in which its gray level lies. However, if the image is noisy, this pixel-by-pixel segmentation process may make many errors, since the noise will cause some of the pixels belonging to one type of region to have gray levels lying in the intervals corresponding to another type. Segmentation could become more reliable if we first smoothed the image to reduce its noisiness.

An image can be smoothed by local averaging, i.e., averaging the gray level of each pixel with the gray levels of a set of its neighbors. However, this process will blur the boundaries between the regions, since a pixel near such a boundary has neighbors lying in both its own region and the adjacent region. If we knew which neighbors belonged to the same region as the pixel, we could use only these neighbors in the average. In other words, the quality of the smoothing process would be improved if we could first segment the image into the appropriate regions, so that smoothing could be performed within the regions only, not across their borders.

These remarks suggest that it might be preferable to perform smoothing and segmentation concurrently, using some type of cooperative process. An example is the combined smoothing and neighbor linking process defined in [1]. Here weights are assigned to the links between a pixel and its neighbors based on their similarity; the image is smoothed by weighted averaging of each pixel with its highest-weighted neighbors; concurrently, the weights are adjusted as the similarities between neighbors change. The process is iterated, with weighted averaging and weight adjustment alternating. Note that this process does not involve classification of the pixels, but does yield a segmentation of the image into regions based on the connectedness relation defined by the links, if we threshold their weights.

This paper deals with another approach to concurrent smoothing and segmentation based on linking, using versions of the image at different resolutions and defining links between overlapping "pixels" at successive resolutions. In a low-resolution image, the pixels interior to regions have gray levels that are less noisy, since a pixel at low resolution represents an average and is thus less variable. On the other hand, the lower the resolution, the less likely it is that a pixel is contained in a single region; most pixels will overlap two or more regions. The approach considered here, which was first described in [2], takes advantage of both high and low resolutions by using a cooperative process in which the images of successive resolutions interact. A detailed description of this approach will be given in Section 2. A number of variations on the multilevel approach have been investigated, involving changes in the initialization of the process, the method of defining links, and the iteration sequencing; these are described in Section 3.

2. Multiresolution pixel linking

Let the size of the original image be 2^n by 2^n. To define the reduced-resolution versions of the image, we make use of an exponentially tapering "pyramid" of arrays of sizes 2^{n-1} by 2^{n-1}, 2^{n-2} by 2^{n-2}, ..., 4 by 4, 2 by 2, so that the kth level has size 2^{n-k} by 2^{n-k}. To avoid border effects, all these arrays are regarded as cyclically closed, i.e., the first column is regarded as lying to the right of the last column, and the top row below the bottom row. The elements of each array will be called pixels or nodes. Many different schemes can be defined for constructing such pyramids [3], but in our experiments we used only the simple scheme that will now be described.

We will assign gray levels to the nodes at each level ($k>0$) by taking (weighted) averages of the gray levels of 4-by-4 blocks of nodes at the level below it. The blocks corresponding to adjacent nodes overlap by 50%; this is why the reduction in size from level to level is by a factor of 2, not a factor of 4. For example, suppose node (i,j) at level $k>0$ corresponds to the block of nodes

$$\begin{array}{cccc}
(u,v) & (u+1,v) & (u+2,v) & (u+3,v) \\
(u,v-1) & (u+1,v-1) & (u+2,v-1) & (u+3,v-1) \\
(u,v-2) & (u+1,v-2) & (u+2,v-2) & (u+3,v-2) \\
(u,v-3) & (u+1,v-3) & (u+2,v-3) & (u+3,v-3)
\end{array}$$

at level $k-1$ (where $(u,v) = (2i-1,2j+1)$). Then node $(i+1,j)$ corresponds to the block

(u+2,v)	(u+3,v)	(u+4,v)	(u+5,v)
(u+2,v-1)	(u+3,v-1)	(u+4,v-1)	(u+5,v-1)
(u+2,v-2)	(u+3,v-2)	(u+4,v-2)	(u+5,v-2)
(u+2,v-3)	(u+3,v-3)	(u+4,v-3)	(u+5,v-3)

where all additions and subtractions are modulo 2^{k-1}. It is easily seen that any node (u,v) below the top level (i.e., $k<n-1$) belongs to four blocks corresponding to nodes on the level above it - in our example, the nodes (i,j),(i-1,j), (i,j+1), and (i-1,j+1). [Note that only for the last of these nodes does (u,v) belong to the center 2-by-2 portion of its block; for the other three, (u,v) is a border point of their blocks.] The level k-1 nodes in the block corresponding to a given node at level k will be called its *sons*, and the level k nodes to whose blocks a given node at level k-1 belongs will be called its *fathers*. Thus every node at level > 0 has 16 sons, and every node at level < n-1 has four fathers. Note that since there are only 16 nodes at level n-2, each of them is a son of all four nodes at level n-1, so that every node in the pyramid is a descendent of every one of these "top" nodes.

The node linking process is as follows: the reduced resolution images are initially defined by unweighted averaging of the gray levels in each block. The gray level of each node is then compared with the levels of its four fathers, and a link is established between the node and its most similar father, i.e., the father whose level is closest to the node's level. After this has been done at every level, we recompute the gray level of each father by averaging only those sons that are linked to it. (If no sons are linked to a father, we give it "gray level" zero.) Based on these new averages, a node's most similar father may have changed, so we next change the links as necessary, then recompute the averages, then change the links again, and so on. Typically, this process stabilizes after a few iterations.

To see how this process works, let us define the *base* of a node as the set of pixels that are linked (through as many intermediate stages as necessary) to that node. Thus initially the base of every node is a square block of pixels. If the base of a node initially lies mostly inside a region, the node is most likely to become linked to nodes on the level below that also lie (mostly) in that region; thus its recomputed average will become closer to the region average. As the process is iterated, nodes at relatively high levels acquire values that approach the average values of regions, even though they are too large to fit into a region. Slight initial biases in the node averages at high levels will result in high-level nodes being driven toward values that correspond closely with the averages of regions or sets of similar regions in the image. For further discussion of the process, see [2].

Suppose that there are not more than four types of regions in the image. For each type, there should be at least one node at the top level of the pyramid whose average converges to the average gray level of the regions of that type. This node will be linked to nodes which are linked to nodes ... which are linked to the pixels belonging to these regions. In other words, this node becomes the root of a tree whose leaves are the pixels that lie in regions of the given type. If there are fewer than four types of regions, there may be two such trees corresponding to the same region type, representing different subsets of the pixels in these regions. If we know how many region types there are supposed to be, we can suppress some of the nodes at the top level (i.e., forbid anyone to link to them), keeping only as many top-level nodes as there are types. [Alternatively, we can "merge" some of the top-level nodes together, averaging together their values and using this average as the value for each of them.] In this way, we can insure that the number of trees (having distinct values) is the same as the desired number of region types.

In summary, the iterative linking and averaging process is defined as follows:

a) Initialize the node values by simple block averaging of each node's 16 sons

b) Link each node to that one of its four fathers whose value is closest to its own

c) Recompute the node values by averaging the values of only those sons that are linked to the node

d) Change the links in accordance with these new values

e) Repeat steps (c-d) as many times as desired. Typically, there is little change after the first few iterations, and there is no change at all after 10 or 15 iterations.

At any stage of this process, the links define a set of (up to four) trees rooted at the top level of the pyramid, and we associate with each pixel the value at the root of its tree. Thus the process smooths the image to an extreme degree, giving each pixel its tree average as a smoothed gray level. At the same time, it segments the image into (up to four) subsets, where each subset consists of the pixels which are the leaves of one of the trees.

The smoothing and segmentation accomplished by this process can be compared with those achieved by the pixel linking process of [1]. In [1] the links are all at the pixel level, and the smoothing is local. Even if the link strengths all converged to values of 1 (within a region) and 0 (between regions), many iterations would be required to obtain the global average of each region at each pixel of the region, since it takes O(region diameter) iterations for information to propagate across the region. In the process described here, on the other hand, the links are between levels, and information can propagate "across" a region in O(log region diameter) iterations, since nodes comparable in size to the region are only (log region diameter) levels above the pixel level. Moreover, in our process smoothing can take place even over sets of non-connected regions of the same type, whereas the process of [1] can smooth only within a connected region.

The concept of linking each node with its most similar father may be compared with the smoothing processes described in [4-6], where a set of neighborhoods lying on various sides of a pixel are examined, and the pixel's value is replaced by the average of the least variable of these neighborhoods (since this neighborhood presumably lies almost entirely within the pixel's region). Using the most similar neighborhood (i.e., the one whose average is closest to the pixel's gray level), rather than the least variable neighborhood, would probably work well too; but we could not use the least variable neighborhood in our scheme. In any event, the methods of [4-6] use neighborhoods of only a single size, which limits the speed with which the smoothing can propagate, as discussed in the previous paragraph.

3. Variations

In the experiments described in this section, several variations on the basic pyramid linking process were tried. These variations were concerned with how to initialize the node values; how to choose the father to which a node is linked, and in particualr what to do in case of ties; and how the iteration process is sequenced. In the following paragraphs we describe the variations, and then show the results obtained by using combinations of these variations on a standard set of images (which were also used in [2]): an infrared image of a tank, a portion of a blood smear, and a portion of a chromosome spread. These images are shown in Figure 1 (a-c). All results are shown for a stage at which the iteration process has stabilized; this is usually after about 10 iterations.

a) Initialization. In the method used in [2], the value of each node was initialized by averaging the values of all 16 of its sons. An alternative, which (as we shall see) seems to give better results, is to initialize by averaging the values for only four of the sons, namely those whose positions in the image are closest to that of the node. (The position of a node is understood to be at the center of its block.) Note that in this alternative scheme, the intial averages are all nonoverlapping.

b) Father selection. In the method of [2], each node is linked to the father closest in value to the node. A more general idea is to take into account both closeness in value and closeness in position. We can compute link strengths based on a formula such as $\Delta(D+s)$, choosing the father for which $\Delta(D+s)$ is smallest, where Δ is the difference in value, D is the Euclidean distance between positions, and s is a parameter which is used to vary the effect of the D contribution (for large s, differences in D have little effect).

b') Ties. If two fathers have the same link merits, we resolve the tie based on any arbitrary ordering of the fathers, e.g. NW, NE, SE, SW. The choice of this ordering should not significantly affect the results.

c) Sequencing. In [2], links are determined for all levels; then averages are recomputed for all levels; and this process is repeated. An alternative is to iterate level by level: as soon as the links from the nodes at level k are redefined, the averages at level k+1 are recomputed, and the links from level k+1 are then redefined based on these new averages.

d) Top level nodes. The number ($\leq$ 4) of nodes used at the top level should be the same as the desired number of region types - 2 for the tank and chromosomes, 3 for the bloodcells. We can insure that only two or three nodes at the top level are used by initializing the values of the remaining node(s) to a very high number, thus insuring that no nodes will ever link to them. As a refinement, we can fix the top-level nodes that we do use to have values that represent estimates of the expected region averages; we will show some examples using this variation. We will also show examples of results obtained when we use all four nodes at the top level, even though the desired number of region types is less than four. As we shall see, the process then tends to create somewhat artficial discriminations within the regions.

We first show the results obtained when we use the desired number of nodes at the top level, but do not attempt to set the values of those nodes to the expected region averages. Figure 2 (top two rows) shows these results for the four combinations of initialization and sequencing schemes. We see that in the chromosome case (Figure 2c), four-son initialization gives better results; when 16-son initialization is used, some of the small chromosomes are lost, probably because too much of the background is initially averaged with them, so that they link to a top-level node whose value converges to the background value rather than to the chromosome value. The initialization scheme has little effect on the results for the other two images, and the iteration sequencing scheme has little effect on any of the images. The order used for tie-breaking also has little effect, as we see from the bottom left pictures in Figure 2 (which use the same initialization and sequencing schemes as the top left pictures). Finally, the bottom right pictures in Figure 2 show what happens when we give some weight to Euclidean distance (s=5) in choosing the links (otherwise, same as top left); note that this too improves the results in the chromosome case, and has little effect in the other two cases. It seems from these results that four-son initialization is preferable to 16-son initialization, and that it may also be preferable to give some weight to Euclidean distance in choosing links; but the other variations make little difference. The exact shapes of the tank and cell nucleus are somewhat sensitive to variations because the correct links for blocks near the borders of these regions will be somewhat ambiguous, due to the noisiness or texturedness of the regions.

Figure 3 shows analogous results when the top-level nodes are given estimates of the average region gray levels as fixed values. Again, the variations make little difference for the tank and cell images, but they are significant for the chromosome image. The loss of the small chromosome has now become dependent on the iteration sequence and even on the tie-breaking order (!); and when we use the four-son initialization method, a large chromosome is lost (!). Apparently, attempting to fix the values of the top-level nodes as equal to the estimated region averages can actually degrade the performance of the pyramid linking process.

Figure 4 gives analogous results when all four top-level nodes are used, so that the process tries to find four region types in each image.* The resulting artifacts are especially apparent for the chromosome image, where the background gets segmented into three subregions that differ appreciably in average gray level. Here again, when we use 16-son initialization, the small chromosomes become part of the background, but this does not happen when four-son initialization is used, nor when weight is given to Euclidean distance in choosing links. The other variations have little effect, and all of the effects are minor for the cell and tank images (the tank region splits up into "noisy" subregions in various ways, but does not get badly confused with the background). Thus these results support the conclusions derived from Figure 2.

When the process is applied to a perfectly regular input pattern such as a checkerboard, it breaks down and fails to segment the pattern into two region types, unless ties are broken randomly. Figure 5 shows results analogous to those in Figure 4 (left column corresponds to Fig. 4 top left, and right column to bottom right), but using random tie-breaking: the results are quite similar.

*If the input image contains fewer than four gray levels (e.g., if we threshold the chromosome or tank image into two levels or the cell image into three), the process does not create additional values, even though all four top-level nodes are used.

The smoothing effect of the process as we follow the links from level to level can be assessed by constructing histograms corresponding to each level's view of the image. Suppose that, for a given k, we give each pixel a gray level equal to the value of the node at level k to which it is linked. When we do this for k=0, 1, 2, ..., we obtain a sequence of successively smoother and simpler images, whose histograms become successively more spiky, until finally, the histogram obtained from the top level consists of (at most) four spikes. Such histograms for the three images, after one iteration of the linking and reaveraging process, are shown in Figure 6 for levels 0, 1, 2, 3, 4. (16-son initialization was used, and links were chosen based on value similarity only.) If we did not want to rely on the iterative process to converge to a good segmentation, we could still consider using a single iteration of the process to improve the separation of the histogram peaks, so that segmentation by thresholding based on the histogram would be easier.

4. Concluding remarks

This paper has investigated a number of variations on the basic pyramid linking process. The results suggest the following tentative conclusions:

a) It appears to be preferable to use schemes in which some weight is given to the relative positions of nodes, both in initializing their values and in choosing links, especially in cases involving regions that consist of many small connected components. Apparently, when we take relative positions into account, we have a better chance of preserving the integrity of small regions.

b) It is desirable to specify the desired number of region types (i.e., pixel classes), by allowing only that number of nodes at the top level to be "active." Otherwise, the process tends to split some of the classes artificially. On the other hand, using estimates of the average gray levels of the classes to fix the values of the nodes at the top level may degrade the results, perhaps because it introduces premature biases that are not compatible with the early stages of the linking pattern.

c) The process is relatively insensitive to the sequencing of the iterations and to the node ordering used for breaking ties.

The experiments reported in this paper have led to a better understanding of the pyramid linking concept. The conclusions will serve as guidelines in the design of linking processes based on pixel properties other than (average) gray level, for application to the smoothing and segmentation of multispectral or textured images.

References

1. J.O. Eklundh and A. Rosenfeld, Image smoothing based on neighbor linking, IEEET-PAMI 3, 1981, in press.

2. P. Burt, T.H. Hong, and A. Rosenfeld, Segmentation and estimation of image region properties through cooperative hierarchical computation, TR-927, Computer Vision Laboratory, Computer Science Center, University of Maryland, College Park, MD, August 1980.

3. P.J. Burt, Fast hierarchical correlations with Gaussian-like kernels, TR-860, Computer Vision Laboratory, Computer Science Center, University of Maryland, College Park, MD, January 1980.

4. F. Tomita and S. Tsuji, Extraction of multiple regions by smoothing in selected neighborhoods, IEEET-SMC 7, 1977, 107-109.

5. M. Nagao and T. Mutsuyama, Edge preserving smoothing, Computer Graphics Image Processing 9, 1979, 394-407.

6. R.M. Haralick and L. Watson, A facet model for image data, Proc. PRIP 79, 489-497.

Figure Captions

1. The three images used in the experiments: a) tank, b) blood cells, c) chromosomes.

2. Effects of varying the initialization, sequencing, tie-breaking rule, and linking criterion. In all cases, the number of nodes used at the top level is equal to the desired number of region types. Top row: initialization using averages of 16 sons (left: iteration for all levels at once; right: iteration level by level). Middle row: analogous, but initializing using averages of the four sons closest in position. Bottom left: same as top left, but using a different tie-breaking order. Bottom right: same as top left, but using a linking criterion that depends on Euclidean distance as well as on difference in value.

3. Analogous to Figure 2, but initializing the top-level nodes with estimates of the region averages.

4. Analogous to Figure 2, but using all four nodes at the top level.

5. Analogous to the top left and bottom right pictures in Figure 2, but breaking ties randomly.

6. Histograms obtained, after one iteration of linking and re-averaging, when the node values at a given level are assigned to the pixels having those nodes as ancestors, for levels 3 4 .
   0 1 2

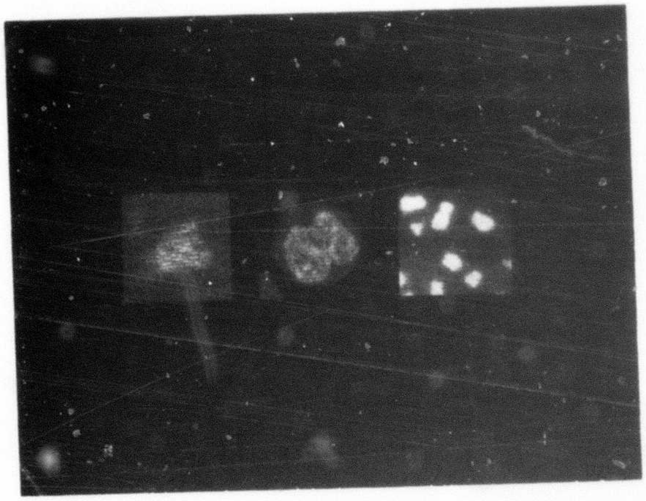

Figure 1

Figures 2, 3, 4: See following pages.

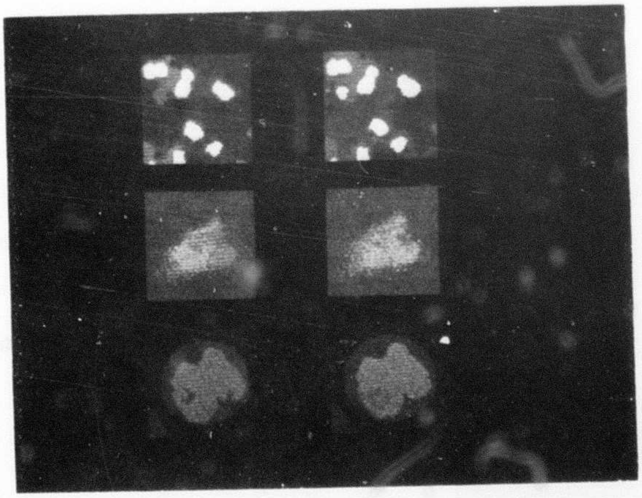

Figure 5

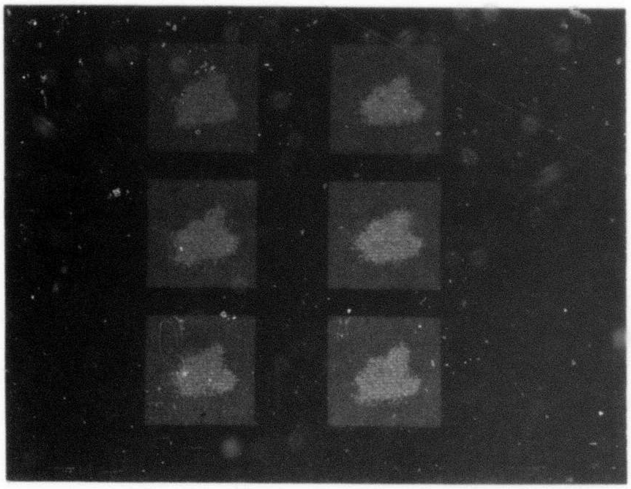

Figure 2a

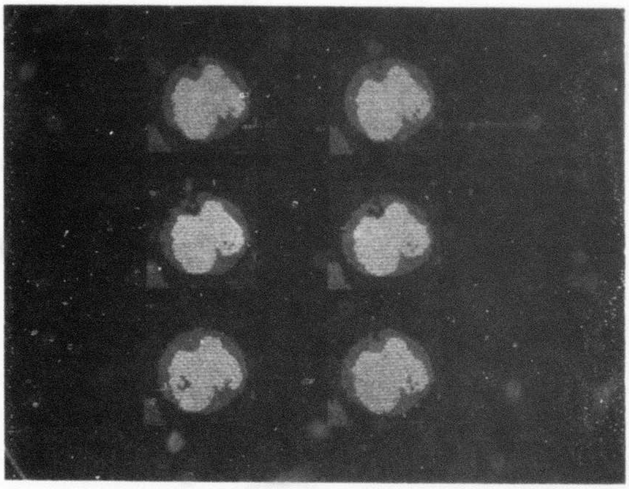

Figure 2b

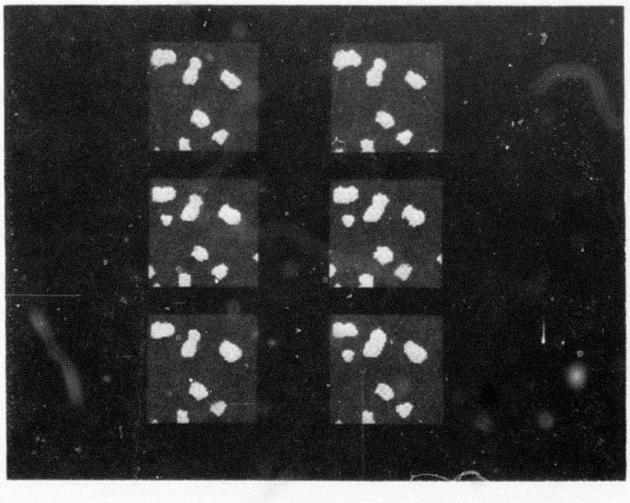

Figure 2c

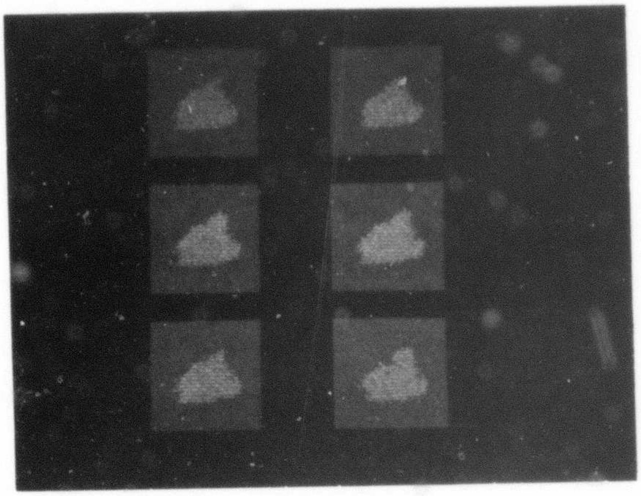

Figure 3a

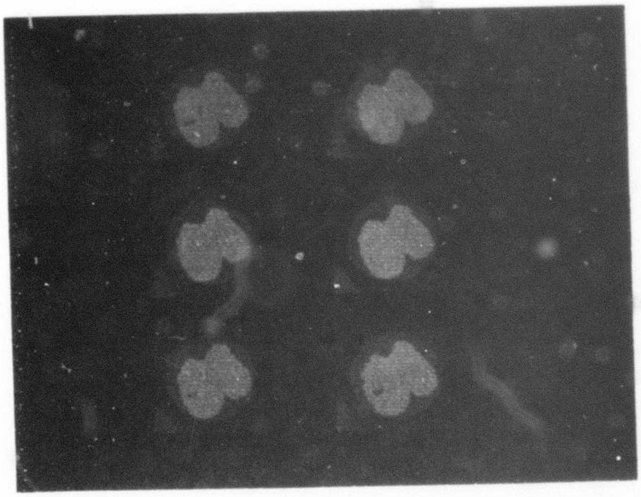

Figure 3b

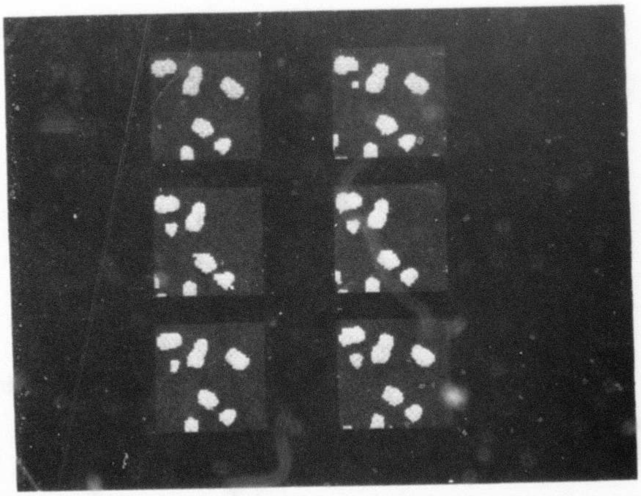

Figure 3c

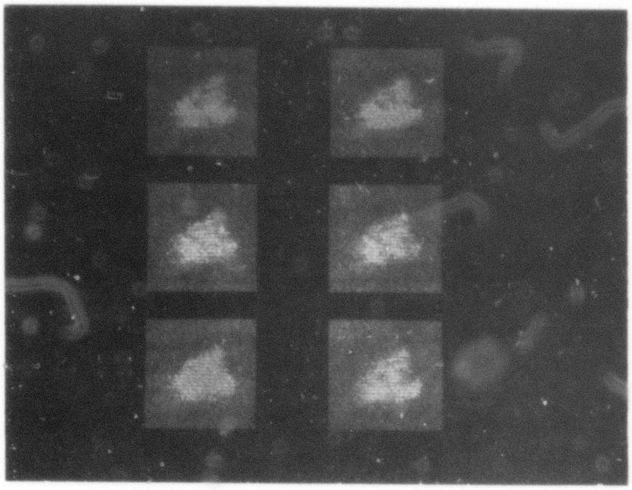

Figure 4a

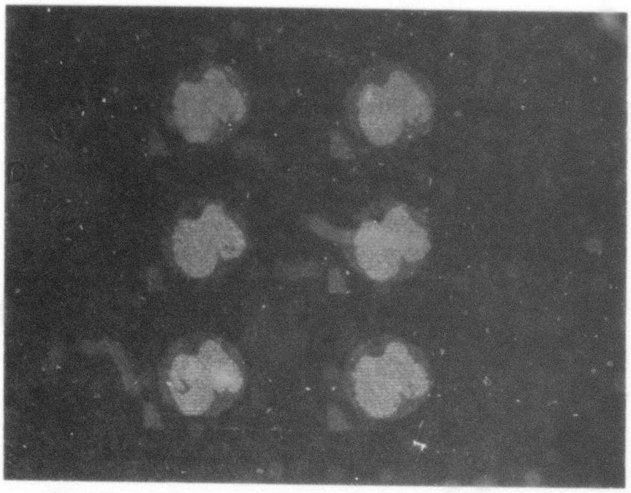

Figure 4b

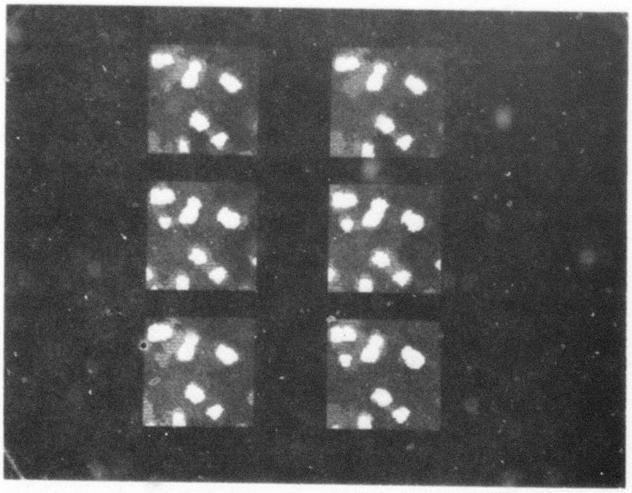

Figure 4c

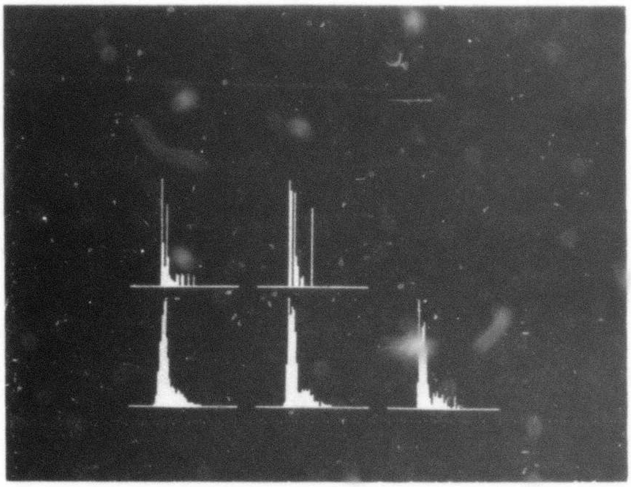

Figure 6a

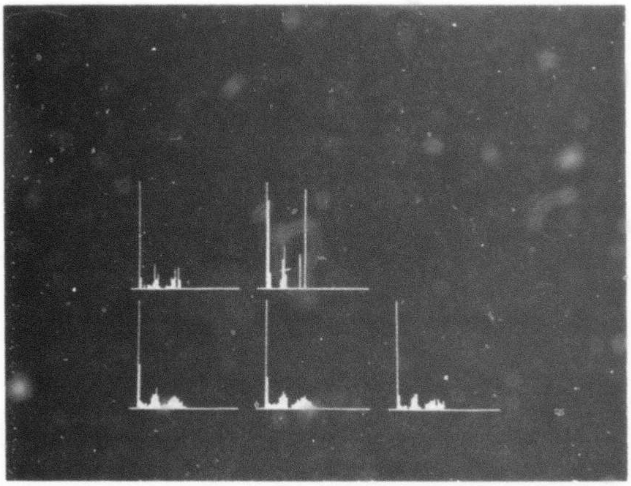

Figure 6b

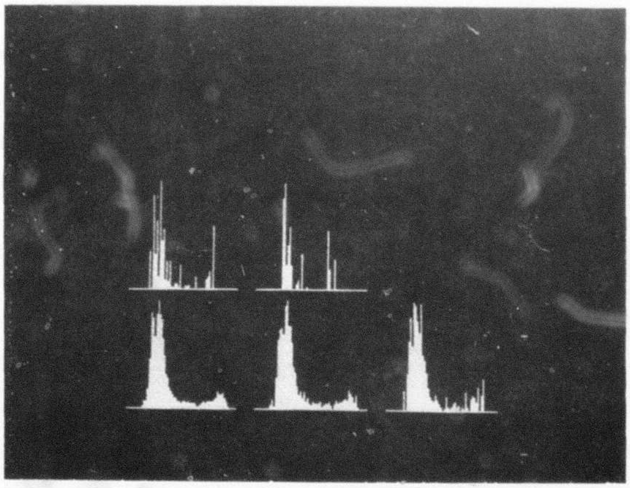

Figure 6c

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS
BEFORE COMPLETING FORM

1. REPORT NUMBER

2. GOVT ACCESSION NO.
AD A109 562

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
MULTI-RESOLUTION PIXEL LINKING
FOR IMAGE SMOOTHING AND SEGMENTATION

5. TYPE OF REPORT & PERIOD COVERED
Technical

6. PERFORMING ORG. REPORT NUMBER
TR-977

7. AUTHOR(s)
Teresa Silberberg
Shmuel Peleg
Azriel Rosenfeld

8. CONTRACT OR GRANT NUMBER(s)
DAAG-53-76C-0138

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Computer Vision Laboratory, Computer
Science Center, University of Maryland,
College Park, MD 20742

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS
U.S. Army Night Vision Lab.
Ft. Belvoir, VA 22060

12. REPORT DATE
November 1980

13. NUMBER OF PAGES

14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)

15. SECURITY CLASS. (of this report)
Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)
Image processing
Image analysis
Segmentation
Smoothing

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

When an image is smoothed using small blocks or neighborhoods, the results may be somewhat unreliable due to the effects of noise on small samples. When larger blocks are used, the samples become more reliable, but they are more likely to be mixed, since a large block will often not be contained in a single region of the image. A compromise approach is to use several block sizes, representing versions of the image at several resolutions, and to carry out the smoothing by means of a cooperative process based on links between blocks of adjacent sizes. These links define "block trees" which segment the image into regions, not necessarily connected, over which smoothing takes place. In this paper, a number of variations on the basic block linking approach are investigated, and some tentative conclusions are drawn regarding preferred methods of initializing the process and of defining the links, yielding improvements over the originally proposed method.